計算機科学実験及演習 ３A
SIMPLEアーキテクチャのプロセッサの実装

京都大学 工学部情報学科 計算機科学コース
計算機科学実験及演習３
HW 担当グループ

2015/04

初めに・・・

- 寝てしまう前に・・・
- 実験ホームページ
  - http://www.kuis.kyoto-u.ac.jp/ecs/isle/
  - 実験を進めるのに必要な資料があります
  - 教官/TA等の連絡先もあります

座学の概要

1. ハードウェア実験
2. SIMPLEアーキテクチャ
3. PowerMedusaボード
4. 実験の進め方

座学の概要

1. ハードウェア実験
2. SIMPLEアーキテクチャ
3. PowerMedusaボード
4. 実験の進め方

ハードウェア実験 - 内容と目的 -

- 内容
  - マイクロプロセッサの方式設計、論理設計
  - FPGA上で応用プログラムを動作
- 目的
  - プロセッサの動作原理を理解する
  - 回路設計、最適化、動作テストの方法を知る
- 参考書
  - 富田眞治、中島浩： コンピュータハードウェア
  - D.A.パターソン、J.L.ヘネシー著、成田光彰訳： コンピュータの構成と設計(上),(下) 第３or４版

ハードウェア実験 - コースでの位置付け -

- ソフト/ハード両方の知識を生かした最適化
  - プロセッサの内部構造を意識したプログラムの最適化
  - プログラムを効率よく実行できるアーキテクチャ
- ソフト/ハードを統合したシステム全体の設計
  - 一昔前
    - ハードウェア：デバイス技術で汎用プロセッサが高速化
    - ソフトウェア：汎用システム上での新しい情報処理技術を開発
  - 現在
    - クロック高速化の限界 ⇒ マルチコア、メニーコアへ
    - モバイル、組込みシステムの普及 ⇒ 専用プロセッサ、専用回路
    - ハードウェア設計環境の進展
      - ソフトと同じように「コンパイル」
      - 何をソフトで、何をハードで実現するか選択

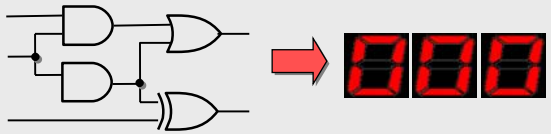
ハード/ソフト両方の知識があるとこの業界では有利
京大計算機コースを出てたらハードもソフトもできて当然

ハードウェア実験の概要

- 実験3ソフトウェアでは高級言語のコードを機械語(ビット列)に変換するものを作る

C言語 `a = b + c;` → アセンブリ `add R1, R2, R3` → 機械語 `01101010...`

- 実験3ハードウェアでは機械語(ビット列)を解釈して実行する論理回路(の塊)を作る

機械語 `01101010...` → 

  - 基本的に、実験2ハードウェアの順序回路の延長

SIMPLEの概要

→Sixteen-bit Microprocessor for Laboratory Experiment

- 😐簡単な命令セット
- ☺基本機能は1通り備えられている

- 特徴
  - １６bit固定長命令
  - ８本の汎用レジスタ
  - １６bit×６４K語の主記憶
  - ロード/ストア・アーキテクチャ
  - ２オペランド形式の命令セット(Rd op Rs -> Rd)

アーキテクチャの説明

- アーキテクチャ
  - コンピュータ全体の構成
    - プロセッサ、メモリ、I/Oなど
  - 主記憶とレジスタの構成ここに含む
- 命令セット・アーキテクチャ
  - 命令の構成
  - 前述のロード／ストア・アーキテクチャは命令セットの形式の1つ
- マイクロ・アーキテクチャ
  - アーキテクチャの回路レベルでの実装

主記憶とレジスタ

コンピュータの状態を表すもの

1. 主記憶
   - 16bit×64K語 （語アドレス方式）
   - ただし、実験で使用するFPGAでは約5K語（旧）、33K語（新）が最大
2. 汎用レジスタ
   - 16bit×8語
3. プログラム・カウンタ (PC)
   - 16bit
4. 条件コード
   - S　サイン
   - Z　ゼロ
   - C　キャリー
   - V　オーバーフロー

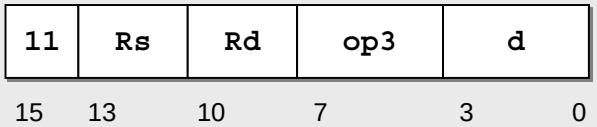
命令セット

コンピュータの状態を変えるもの

1. 演算命令
   - 算術論理演算命令
   - シフト命令
2. ロード／ストア命令
3. 分岐命令
   - 無条件分岐命令
   - 条件分岐命令
4. その他
   - 入出力命令
   - 停止命令

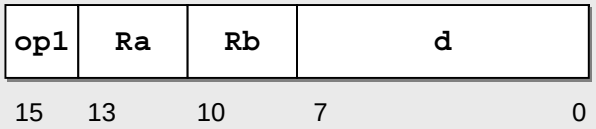
演算命令

- 算術論理演算命令
  - `r[Rd] = r[Rd] op3 r[Rs]`
- シフト命令
  - `r[Rd] = shift_op3(r[Rs], d)`
- 注：実行後に条件コードをセットする



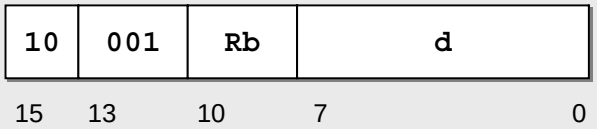
ロード／ストア命令(1)

- ロード命令 (op1 : 00)
  - r[Ra] = *(r[Rb] + sign_ext(d))
- ストア命令 (op1 : 01)
  - *(r[Rb] + sign_ext(d)) = r[Ra]



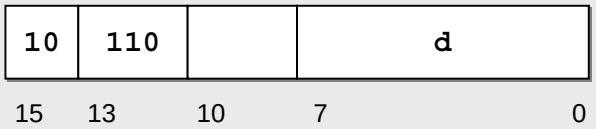
ロード／ストア命令(2)

- 即値ロード命令
  - r[Rb] = sign_ext(d)
  - 即値ロード命令2つとシフト命令で任意の16bitの値をレジスタ格納できる



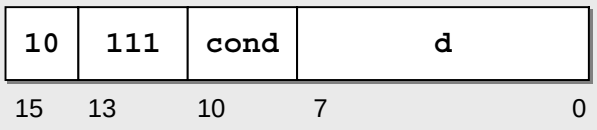
分岐命令(1)

- 無条件分岐命令(B: Branch)
  - PC = PC + 1 + sign_ext(d)



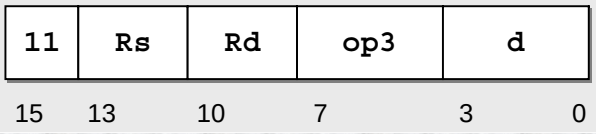
分岐命令(2)

- 条件分岐命令
  - if (cond) PC = PC + 1 + sign_ext(d)
  - 条件コードの値に従って分岐
    - 条件コードは演算命令の実行時にセットされる



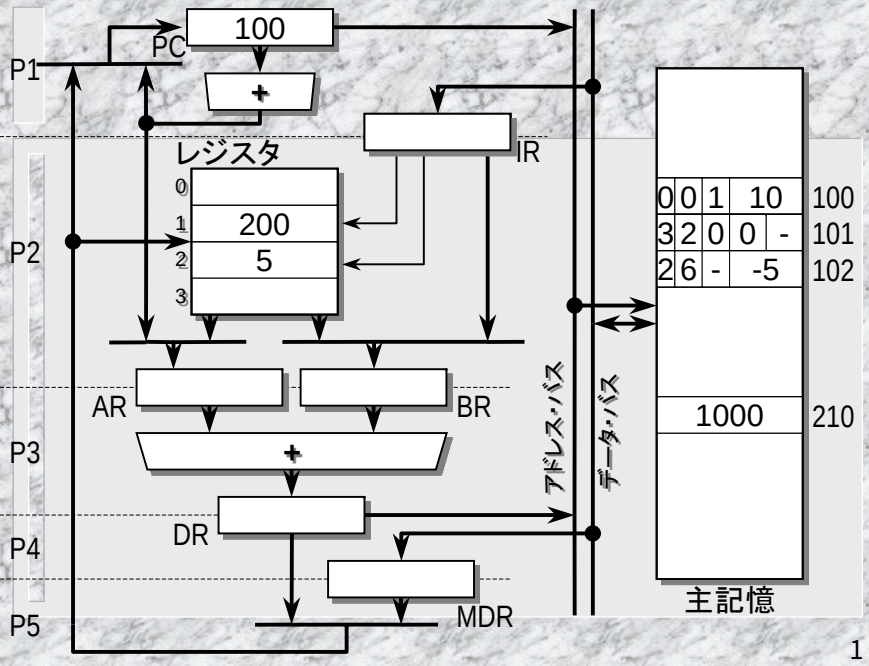
その他の命令

- 停止命令(op3: 1111)
- 入力命令(op3: 1100)
  - r[Rd] = input
  - 入力先はボード上のスイッチ
- 出力命令(op3: 1101)
  - output = r[Rs]
  - 出力先はボードのLED/7SEG LED



基本的な実装　SIMPLE/B

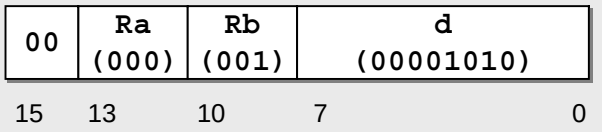
- 次スライドに示すように演算器／レジスタ／データ・パスを配置
- 5つのフェーズを逐次活性化: 実験2の順序回路と同じ
  - P1　命令フェッチ
  - P2　命令デコード、レジスタ読み出し
  - P3　演算
  - P4　主記憶アクセス
  - P5　レジスタ書き込み／PC更新
- フェーズの活性化: 制御部が担当
  - (フェーズへ入力されるデータを保持するレジスタを更新)
  - フェーズ内のセレクタを適切に切り替える
  - フェーズから出力されるデータを保持するレジスタを更新



実行のサンプルの命令

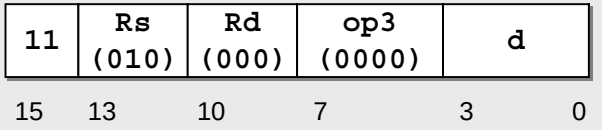
- ロード命令: プログラム・カウンタ100
  - LD R0, 10(R1)

略記 0 0 1 10



- 加算命令：　プログラム・カウンタ101
  - ADD R0, R2

略記 3 2 0 0 -

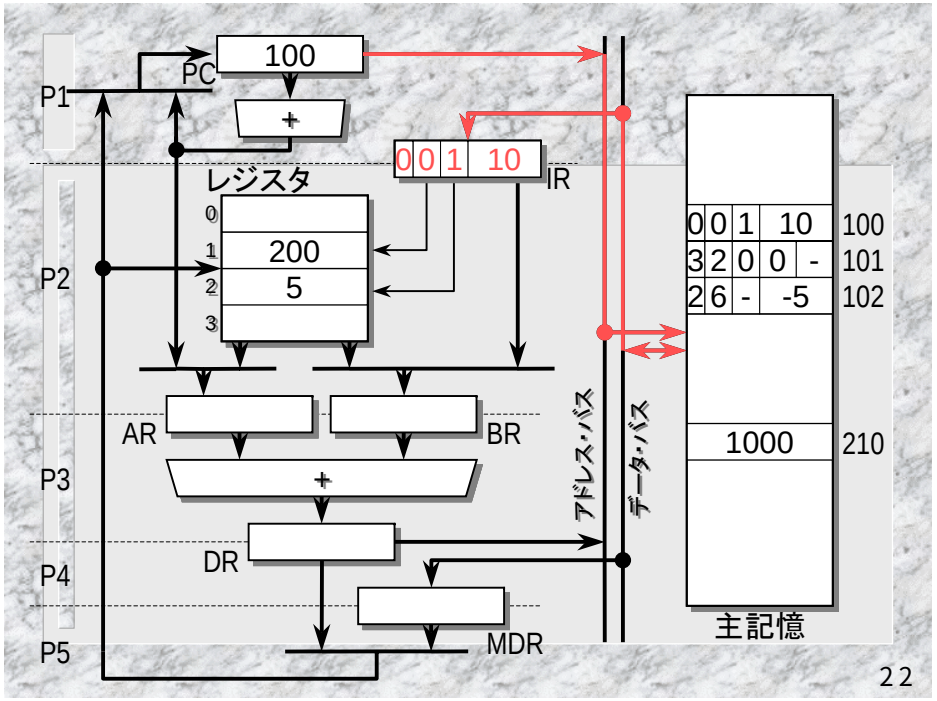


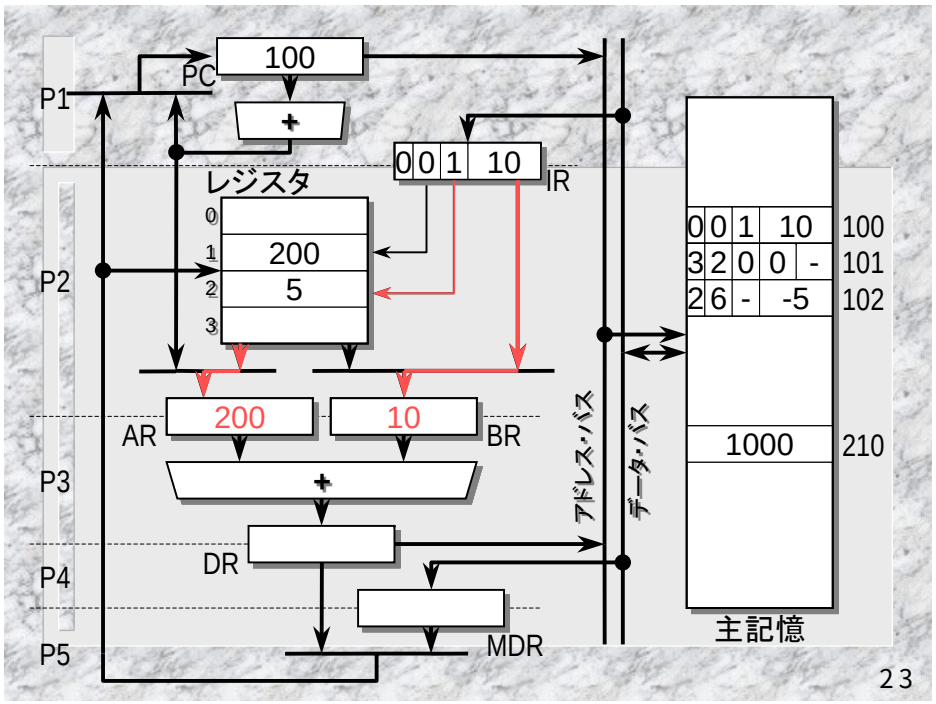
実行のサンプルの命令

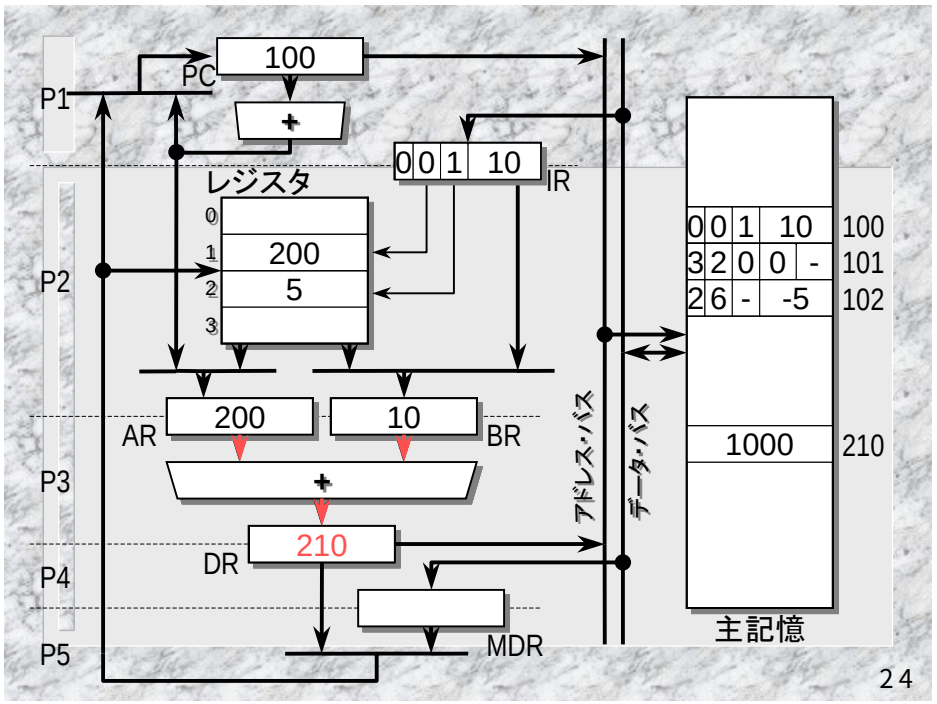
- 無条件分岐命令: プログラム・カウンタ102
  - B -5

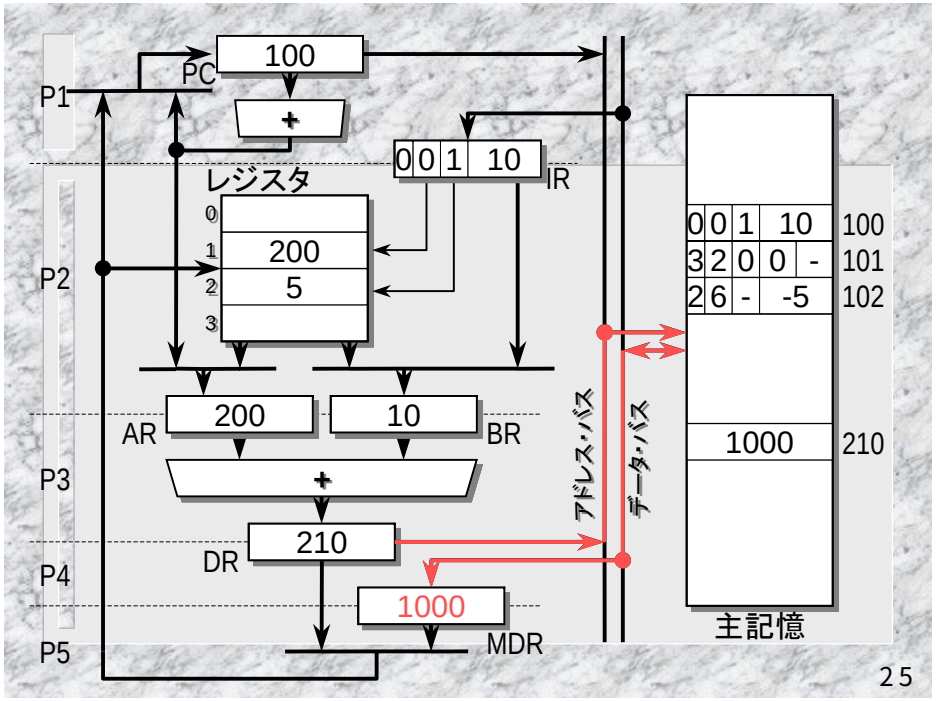
略記 | 2 | 6 | - | -5 |

10	op2 (110)		d (11111011)
15	13	10	7 0

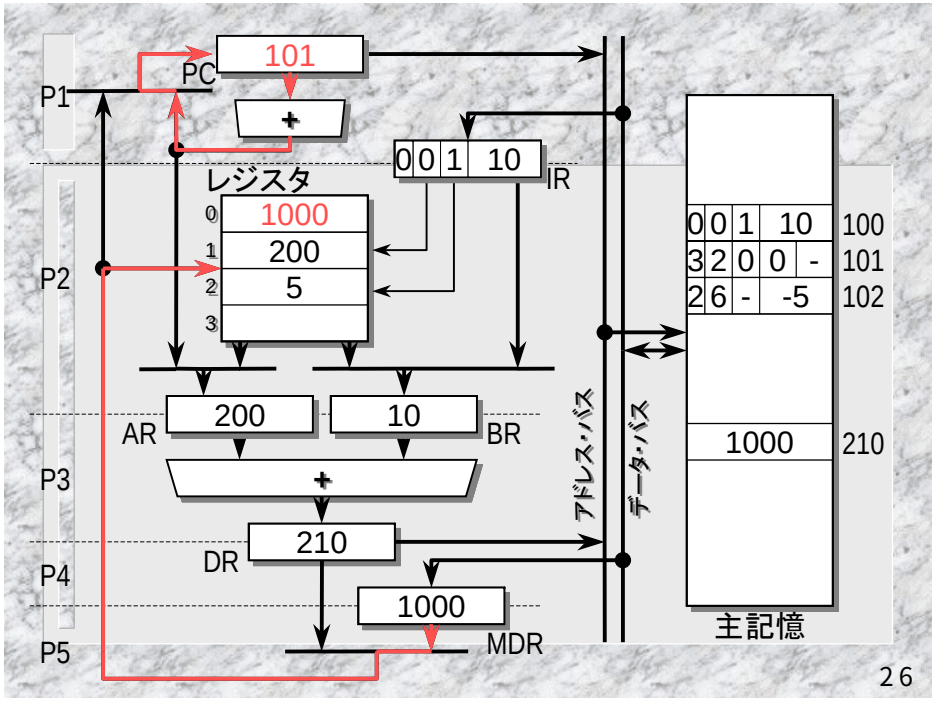








25



26

実行のサンプルの命令

- ロード命令: プログラム・カウンタ100
  - LD R0, 10(R1)

略記 0 0 1 10

00	Ra (000)	Rb (001)	d (00001010)

15 13 10 7 0

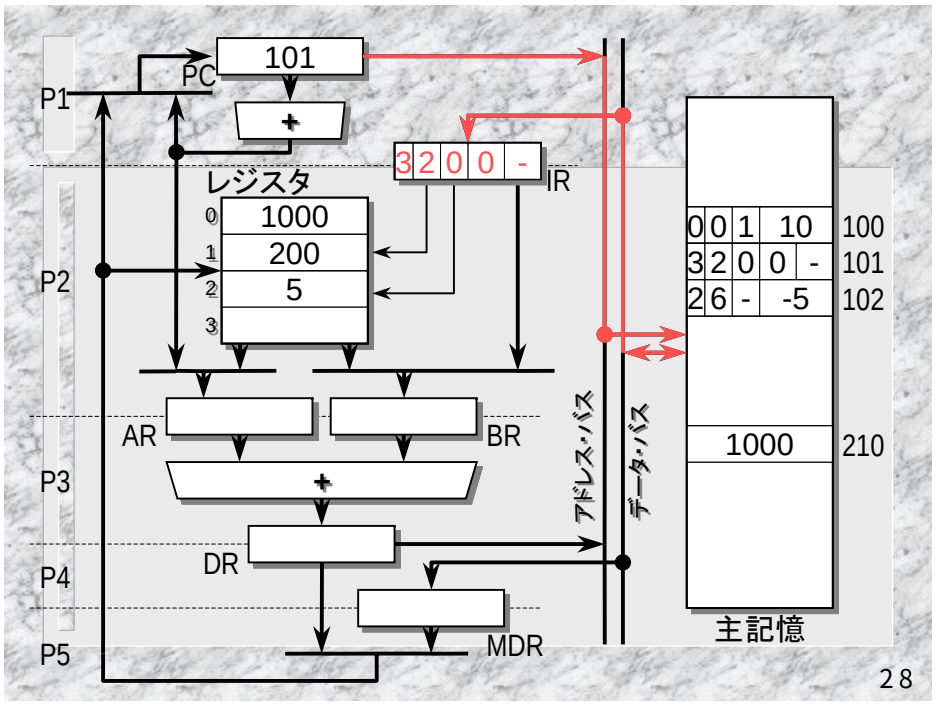
- 加算命令： プログラム・カウンタ101
  - ADD R0, R2

略記 3 2 0 0 -

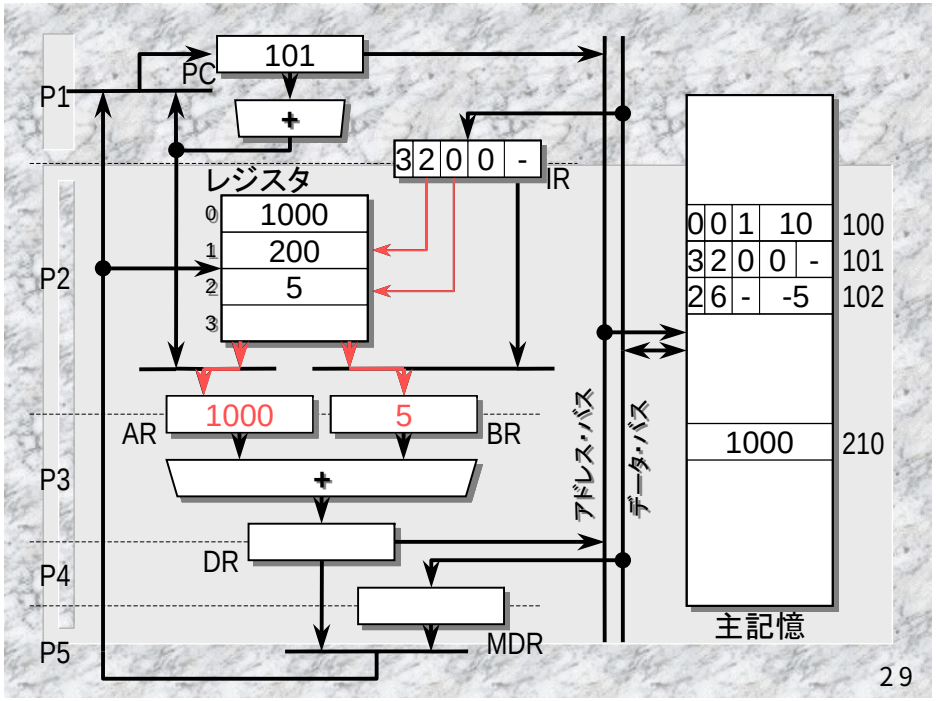
11	Rs (010)	Rd (000)	op3 (0000)	d

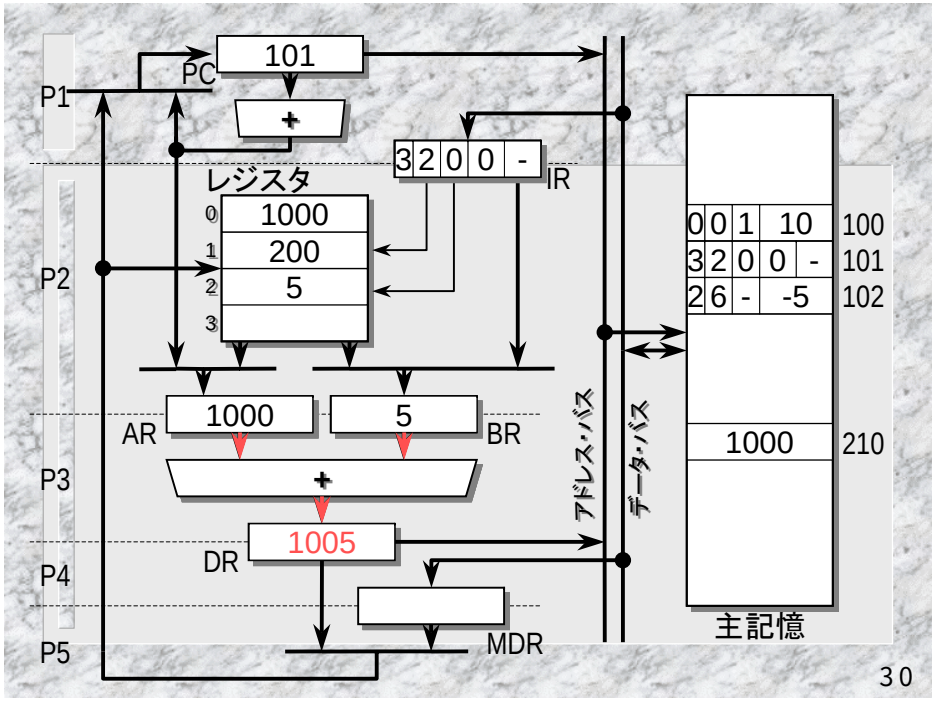
15 13 10 7 3 0

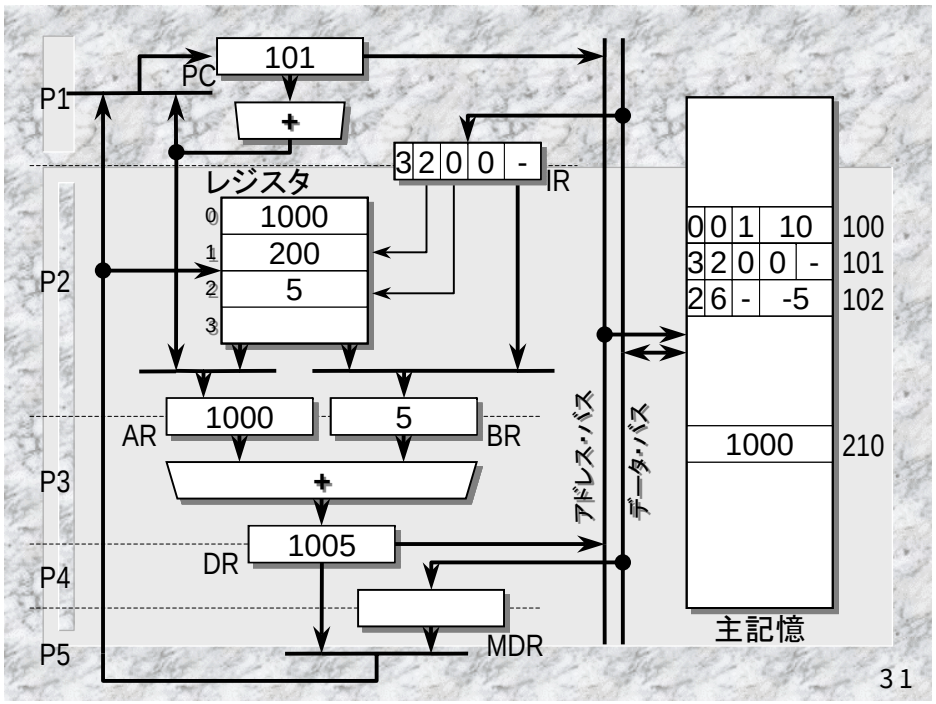
27

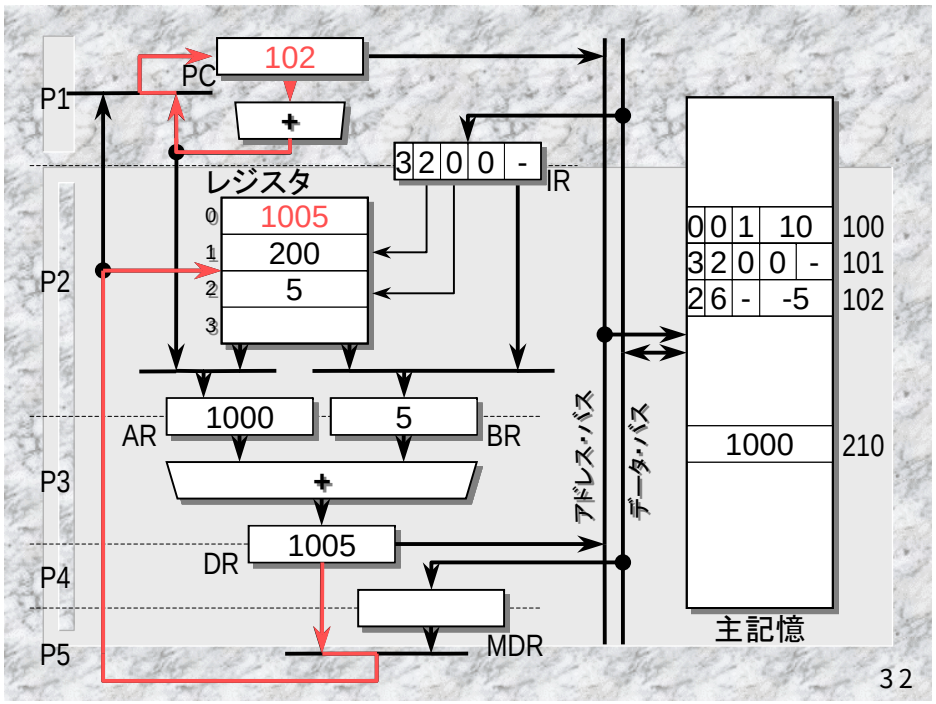


28


101
PC
P1
+
3200-
IR
レジスタ
0 1000
1 200
2 5
3
P2
0010 100
3200- 101
26- -5 102
1000 210
AR 1000
5 BR
P3
+
DR
P4
MDR
P5
アドレス・バス
データ・バス
主記憶
29


101
PC
P1
+
3200-
IR
レジスタ
0 1000
1 200
2 5
3
P2
0010 100
3200- 101
26- -5 102
1000 210
AR 1000
5 BR
P3
+
1005
DR
P4
MDR
P5
アドレス・バス
データ・バス
主記憶
30


101
PC
P1
+
3200-
IR
レジスタ
0 1000
1 200
2 5
3
P2
0010 100
3200- 101
26- -5 102
1000 210
AR 1000
5 BR
P3
+
1005
DR
P4
MDR
P5
アドレス・バス
データ・バス
主記憶
31

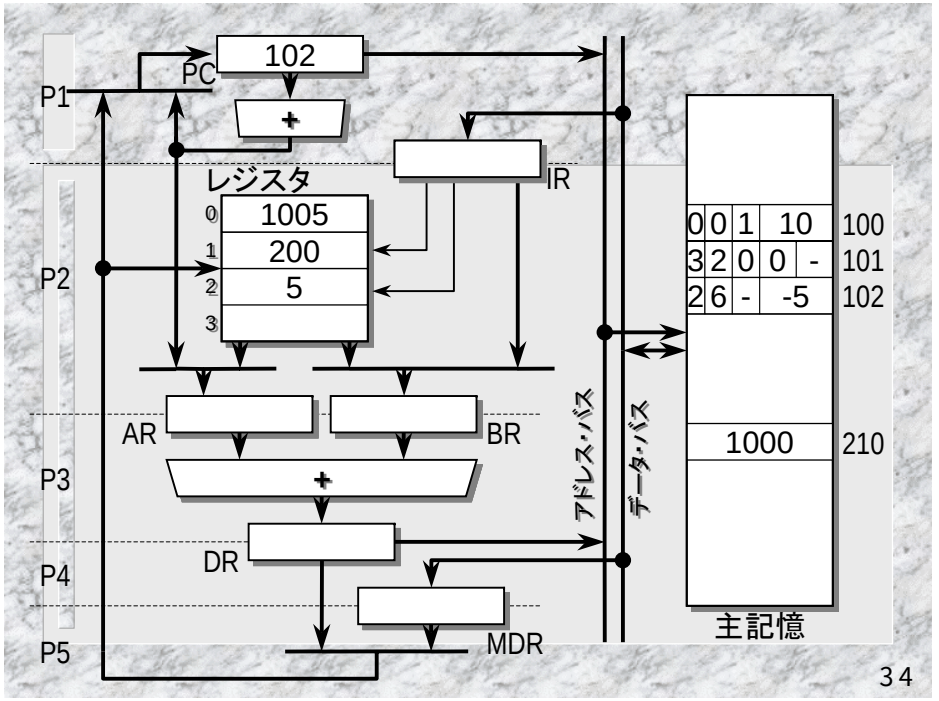

102
PC
P1
+
3200-
IR
レジスタ
0 1005
1 200
2 5
3
P2
0010 100
3200- 101
26- -5 102
1000 210
AR 1000
5 BR
P3
+
1005
DR
P4
MDR
P5
アドレス・バス
データ・バス
主記憶
32

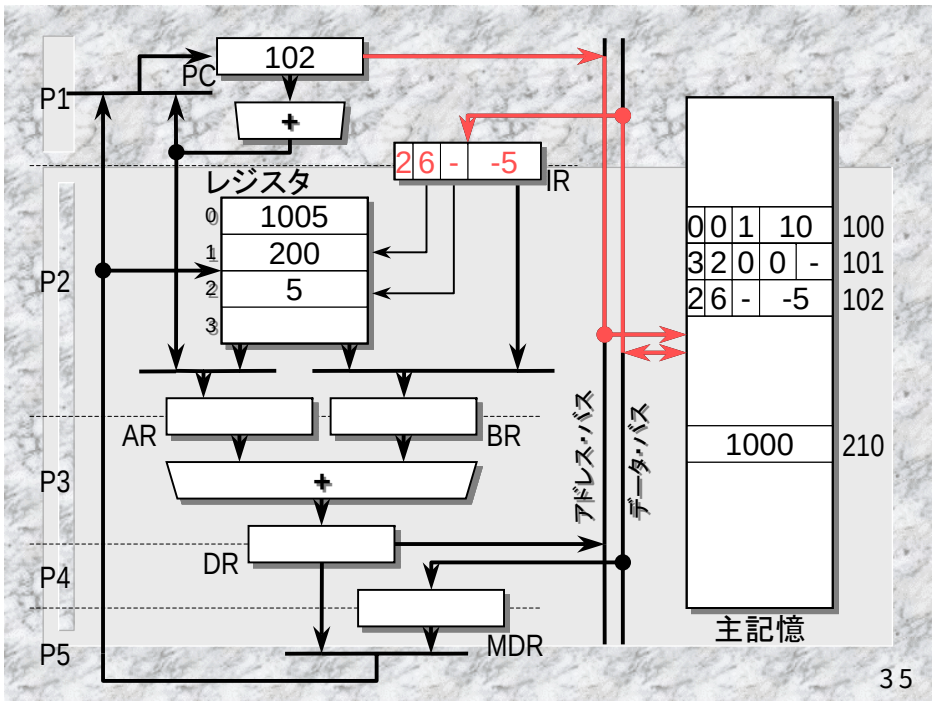
実行のサンプルの命令

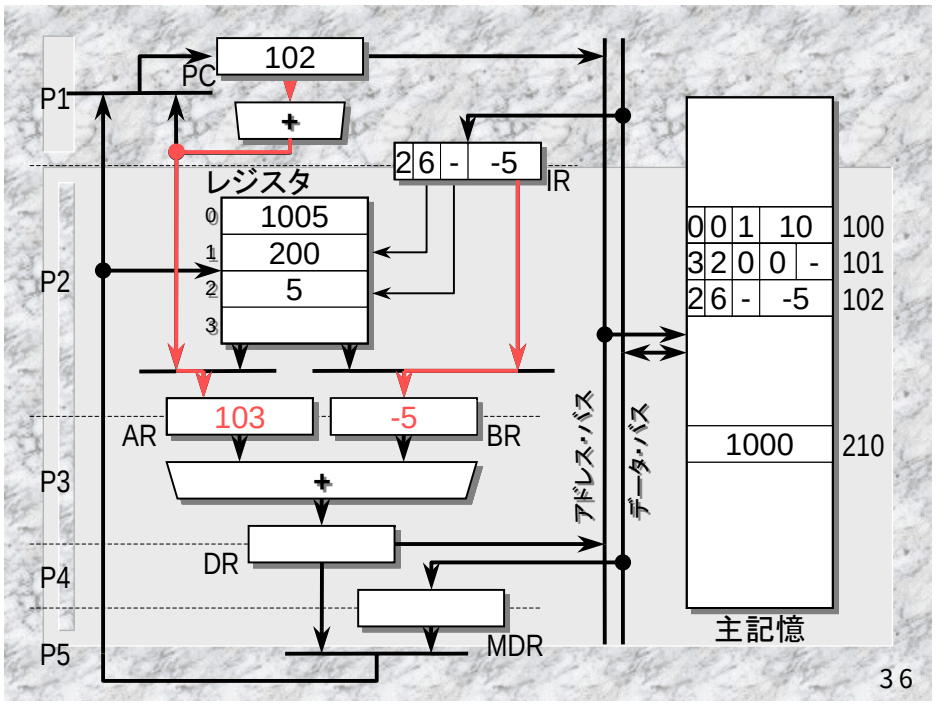
- 無条件分岐命令: プログラム・カウンタ102
  - B -5

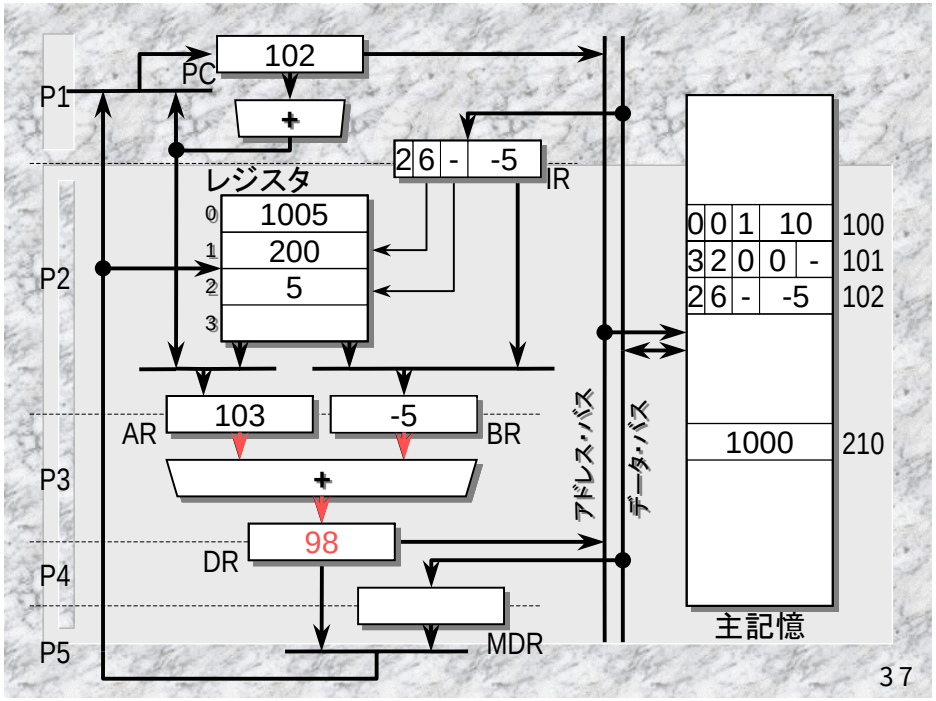
略記 | 2 | 6 | - | -5 |

10	op2 (110)		d (11111011)
15	13	10	7 0

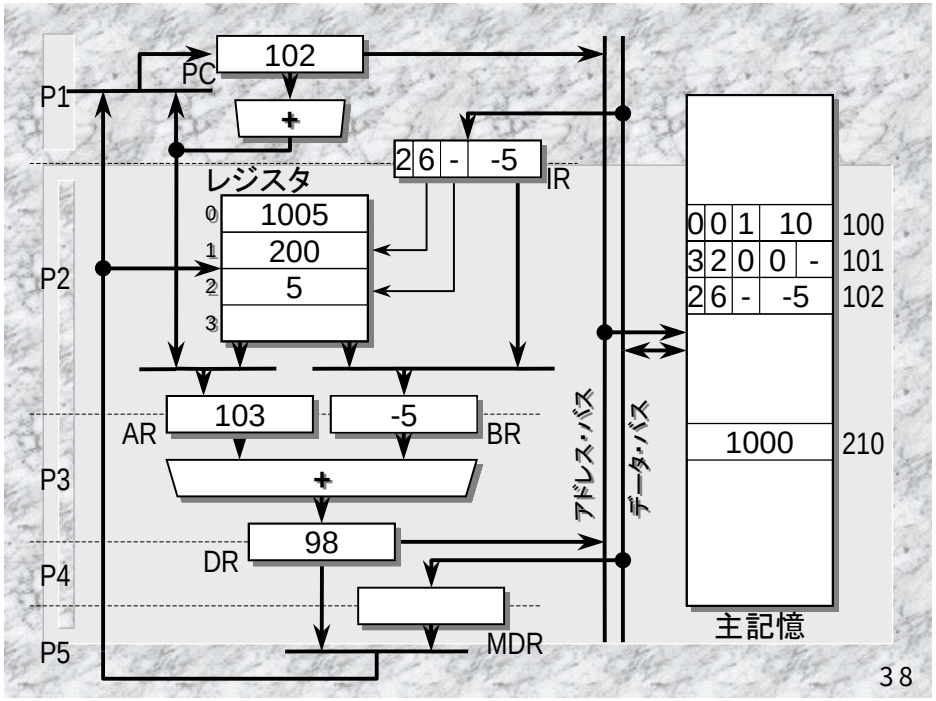




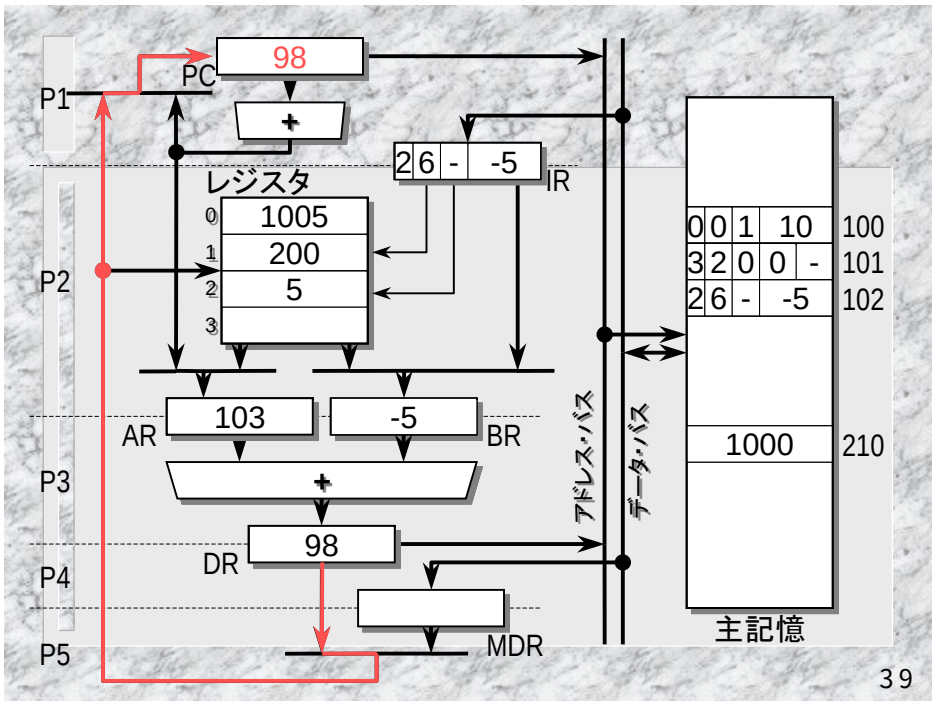




37



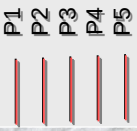
38



39

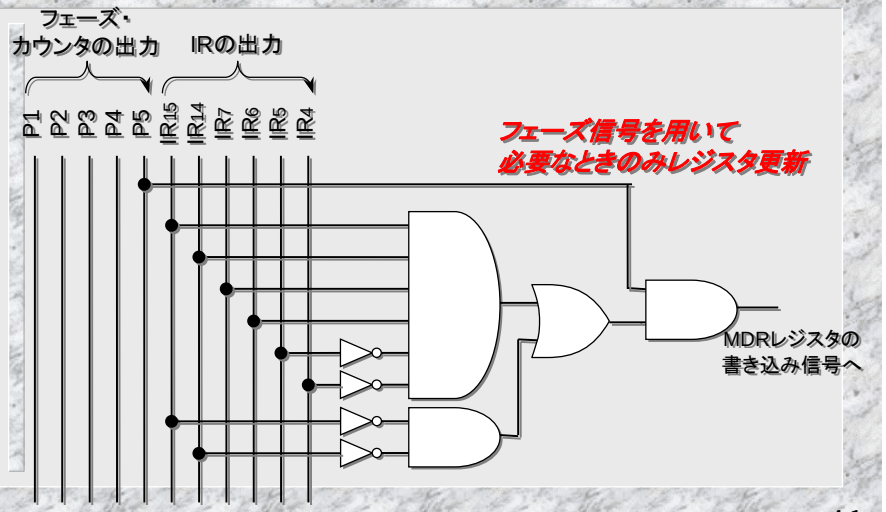
制御部について

- 各種レジスタの書き込み信号やセレクタの切り替えは何がやっている？ -> 制御部がやっている
  - フェーズ・カウンタでフェーズを数える
  - フェーズと命令レジスタの内容により、必要な信号線をアサートする
  - reset信号による各部のリセット、exec信号による各部の動作開始、等
- とりあえず、フェーズを数えるフェーズ・カウンタがあると楽なように設計してあります
  - 各フェーズに対応した信号線をアサート
  - **PC,AR,BR,DR**等はフェーズの信号でアサート

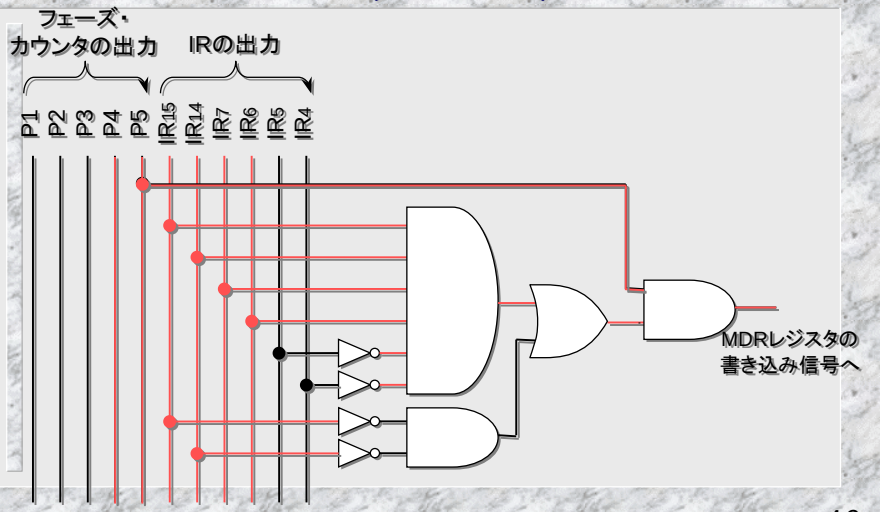


40

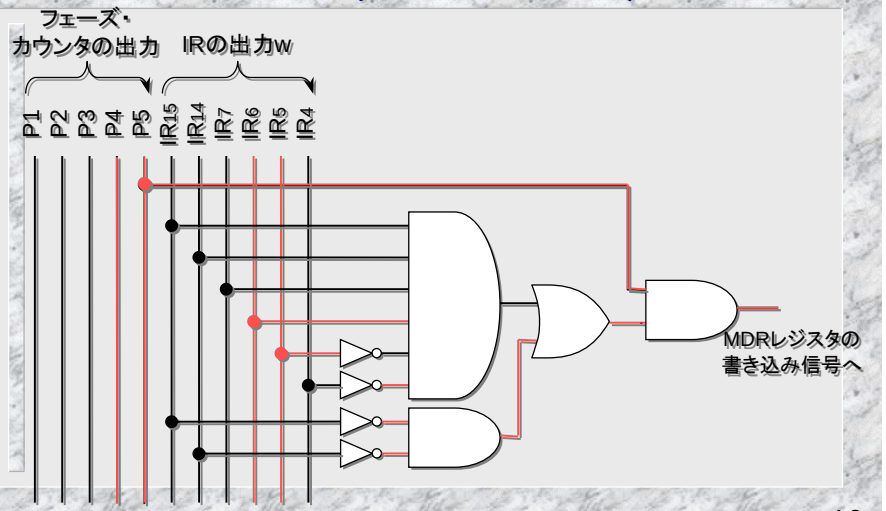
制御信号の例: MDRレジスタ書き込み信号



MDRレジスタ 書き込み信号 (IN命令)



MDRレジスタ 書き込み信号(ロード命令)

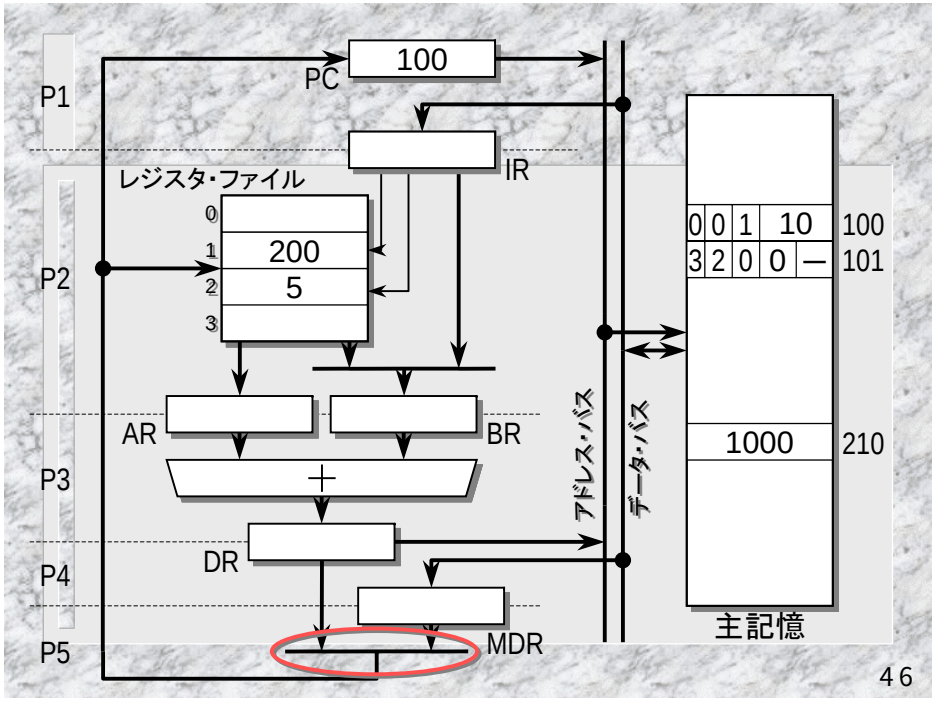


制御部の作り方

- 作るのに最も苦労する所だと思います
- 作り方
  1. 1つの制御信号に対し、IR(の一部)とフェーズに対する真理値表を作る
  2. 最小化や式変形で論理関数を求める
  3. 全ての制御信号に対して1,2を行う
  4. 実装する
- 注意： 制御部は小さくなるように工夫する
  - 小さいほうがミスがまぎれこむ可能性が減る
  - レジスタ番号や演算の指定部は、制御部を通さずに直接必要なところに導く
    - 制御部はできるかぎり簡単になるように命令セットは作られている

セレクタの切り替え信号の例

- ほとんどのセレクタ切り替え信号は組み合わせ回路になります
  - IRの値に応じて切り替え信号を生成すれば良い
- 例：結果値選択信号
  - MDRの値を選択： IN命令、ロード命令
  - DRの値を選択： その他の命令
    - 書き込み先はまた別の制御



セレクタの選択信号の例: レジスタ書き込み値選択信号

- 以下のor
  - 命令がIN命令: IRの上位2bitが11かつop3が1100

11	Rs	Rd	op3 (1100)	d

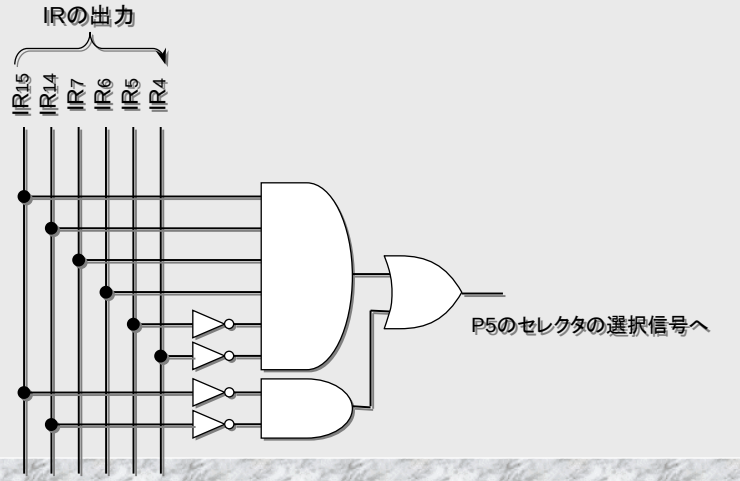
15 13 10 7 3 0

  - 命令がロード命令: IRの上位2bitが00

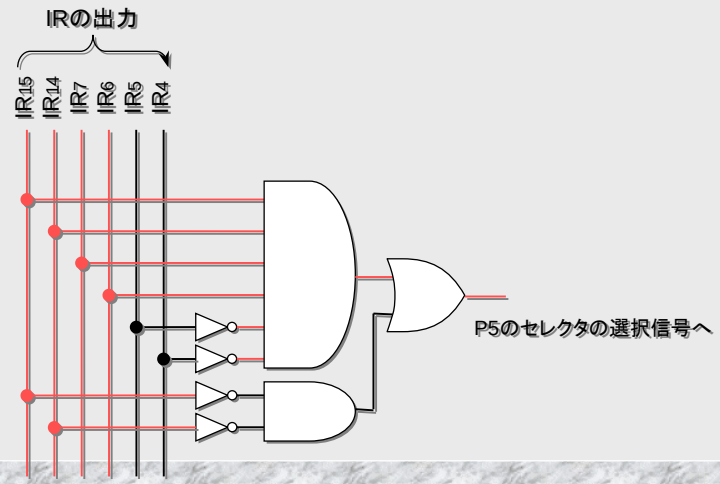
00	Ra	Rb	d

15 13 10 7 0

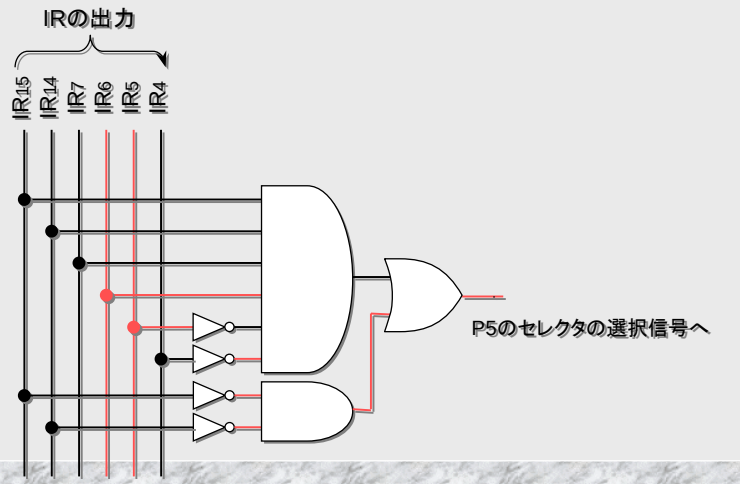
結果値選択信号: 組み合わせ論理



結果値選択信号:
組み合わせ論理(IN命令)



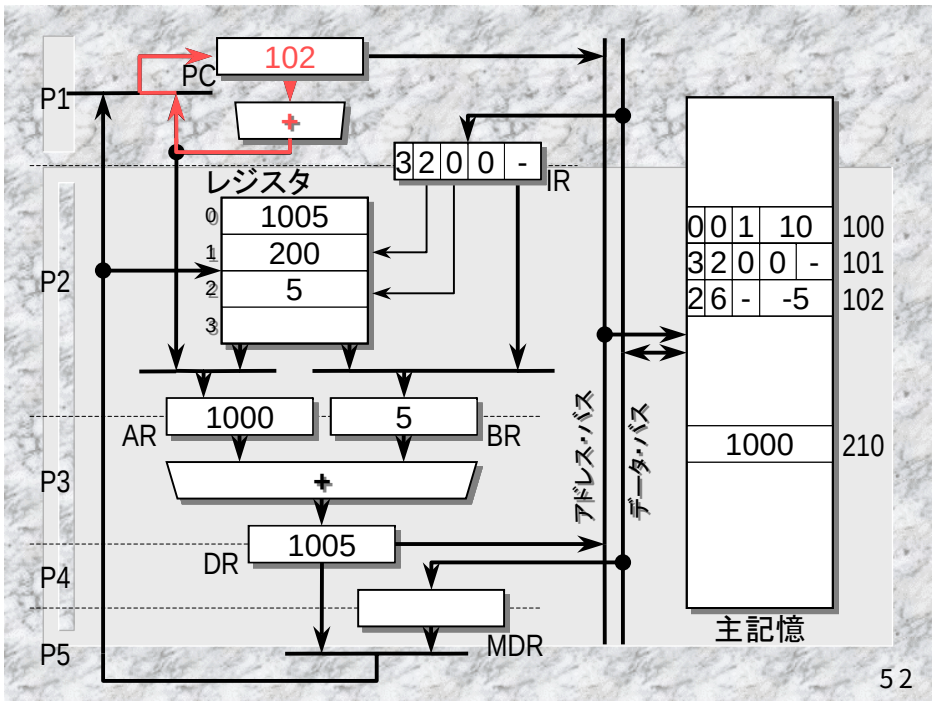
結果値選択信号:
組み合わせ論理(ロード命令)

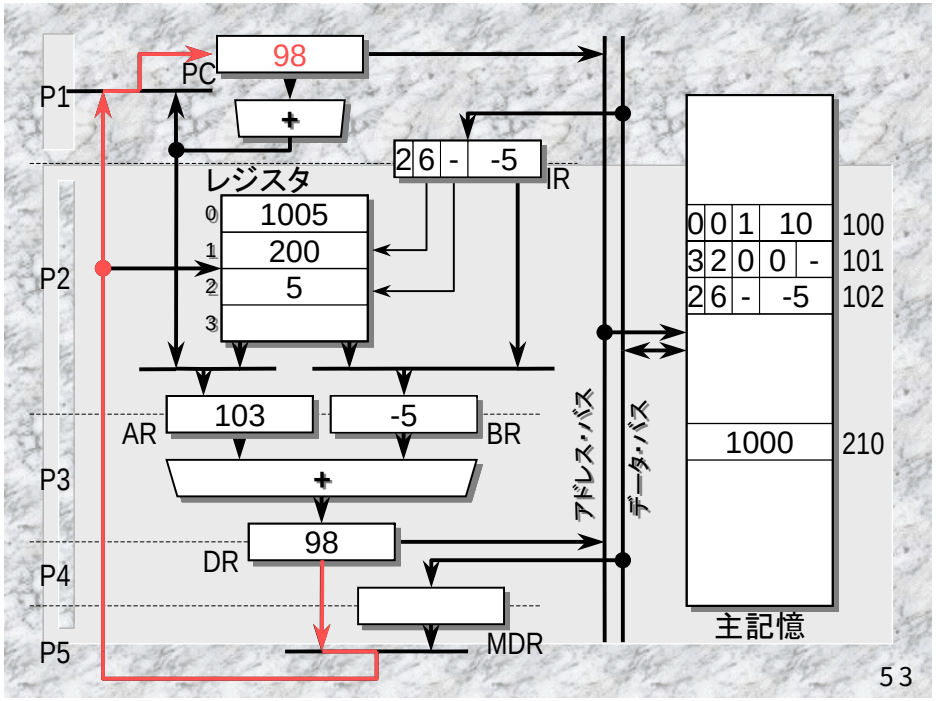


論理設計の例 PC周辺

- PCの動作
  - リセットで0に初期化
  - マルチプレクサで以下の更新値を切り替え
    - PC+1
    - DR

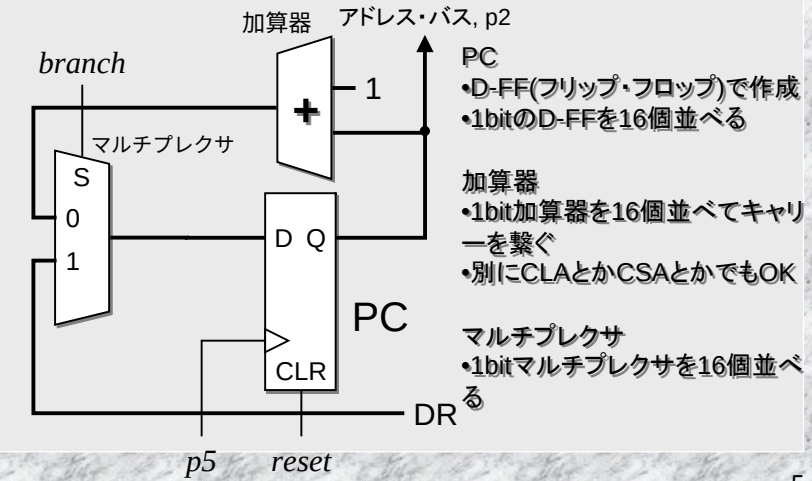
    切替条件は分岐命令＆分岐が成立したか
  - P5で書き込み





53

論理設計の例 PC周辺



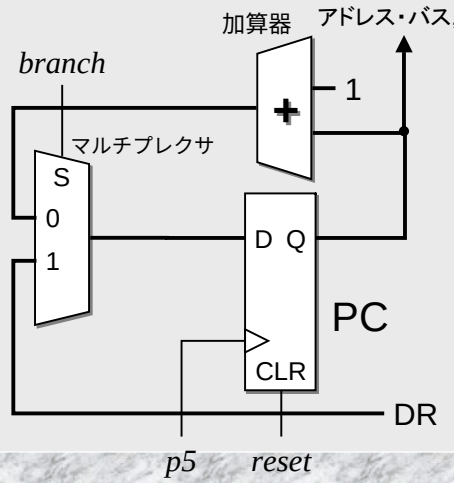
PC
- D-FF(フリップ・フロップ)で作成
- 1bitのD-FFを16個並べる

加算器
- 1bit加算器を16個並べてキャリーを繋ぐ
- 別にCLAとかCSAとかでもOK

マルチプレクサ
- 1bitマルチプレクサを16個並べる

54

論理設計の例 PC周辺



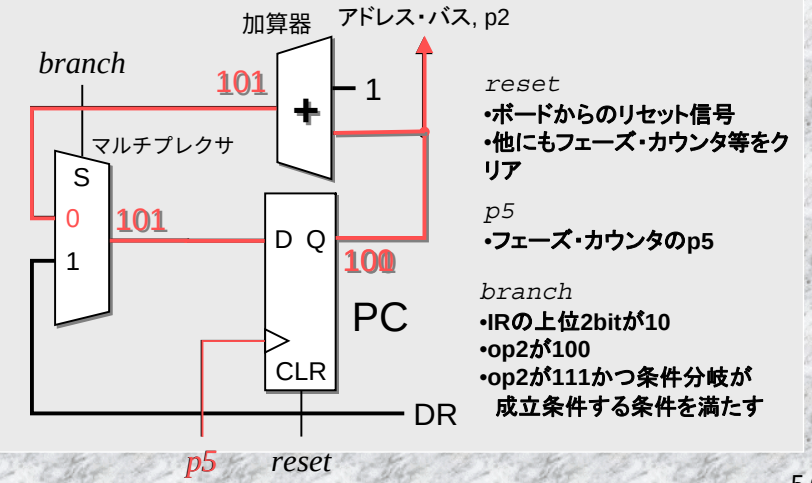
reset
- ボードからのリセット信号
- 他にもフェーズ・カウンタ等をクリア

p5
- フェーズ・カウンタのp5

branch
- IRの上位2bitが10
- op2が100
- op2が111かつ条件分岐が成立条件する条件を満たす

55

論理設計の例
PC周辺(p5, !branchの動作)



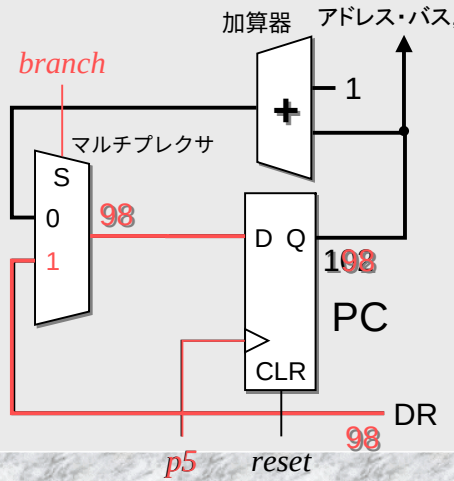
reset
- ボードからのリセット信号
- 他にもフェーズ・カウンタ等をクリア

p5
- フェーズ・カウンタのp5

branch
- IRの上位2bitが10
- op2が100
- op2が111かつ条件分岐が成立条件する条件を満たす

56

論理設計の例
PC周辺(p5, branchの動作)



reset
- ボードからのリセット信号
- 他にもフェーズ・カウンタ等をクリア

p5
- フェーズ・カウンタのp5

branch
- IRの上位2bitが10
- op2が100
- op2が111かつ条件分岐が成立条件する条件を満たす

改良

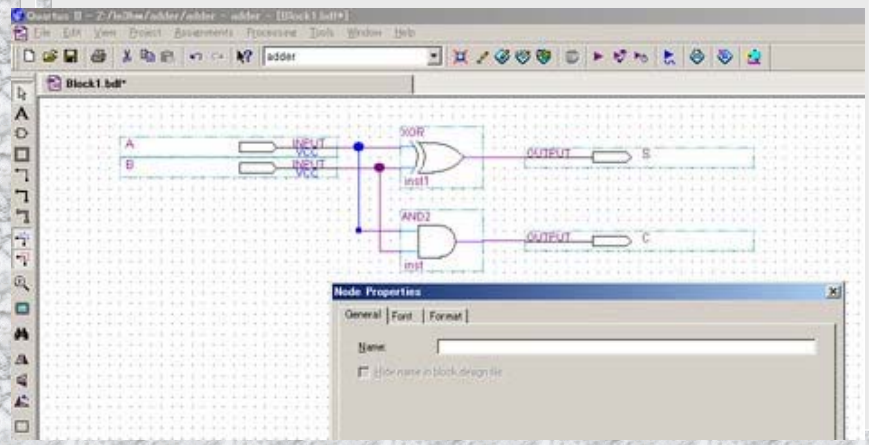
- 命令セット・アーキテクチャの改良
  - 即値オペランドの強化
  - 入出力命令の強化
  - 割り込みのサポート
  - オリジナルの命令の追加
- マイクロ・アーキテクチャの改良
  - フェーズの並列実行(パイプライン化)
- 何らかの拡張を行って、拡張前と比較する
  - プログラムの実行がどれだけ高速化したか？
  - 追加で必要となったハードウェアは？

座学の概要

1. ハードウェア実験の意味
2. SIMPLEアーキテクチャ
3. PowerMedusaボード
   - CADによる設計
   - FPGAとは
   - PowerMedusaボードの概要
4. その他

CADによる設計

- Altera社のQuartus IIを使います

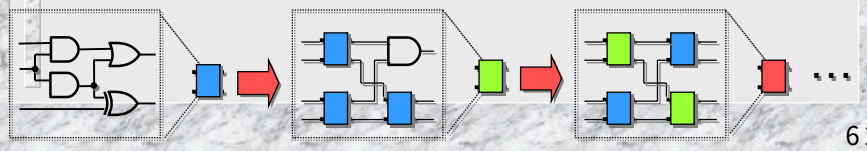


CADの機能

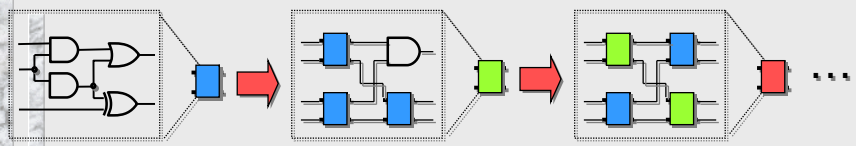
- 設計
  - 基本はブロック・ダイヤグラム・エディタによる設計
  - HDL(Hardware Description Language)も使用可
    - グループ内で合意をとること
    - 一部のモジュールをHDLで設計、など
- 検証
  - 論理シミュレーション
  - タイミング・シミュレーション
- その他
  - メモリ・エディタなど

CADによるハードウェア設計の概要

- 基本的にC言語によるプログラミングと同じ
  - ハードウェアを機能ごとにブロックに分割する
    (＝プログラムを機能ごとに関数に分割する)
    - ブロックへの入力／出力が煩雑にならないように分割
  - 各ブロックの回路を組み、入出力を設定
    - 必要があれば、シミュレータでテスト
  - 作成したブロックを組み合わせて新たなブロックを組む
  - 最上位のブロックの入出力にFPGAのピンを割り当てる



CADによるハードウェア設計の概要



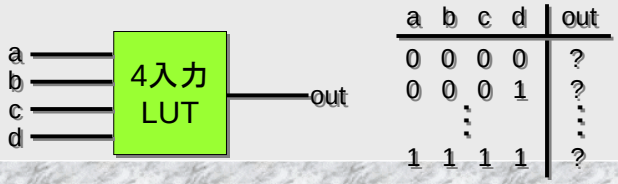
- 設計の例: 16bit桁上げ伝播加算器
  - AND/OR/XORゲートで半加算器を作る
  - 半加算器とANDゲートで全加算器を作る
  - 全加算器を16個ならべ、キャリーを繋げる
- スライド説明の後に、実際にCADを使って回路を作るデモを見せます
  - 回路作成～論理シミュレーション～FPGAボードへのダウンロード

FPGAとは

- FPGA (Field Programmable Gate Array)
  - プログラム可能な LSI の一種
  - 簡単な論理回路を実現できる論理ブロックとそれをつなぐ配線がつめこまれている
  - CADで作成した接続情報をダウンロードして任意の論理回路を実現する

FPGA内部での論理回路の実現

- FPGA内部での論理回路実現方法は様々
  - n入力ルックアップ・テーブル(LUT)
  - 積項型
- 現在はLUTが主流
  - 任意のn入力に対して任意の出力関数を設定可能
  - 論理回路のLUTへの割り付けはコンパイラが行う



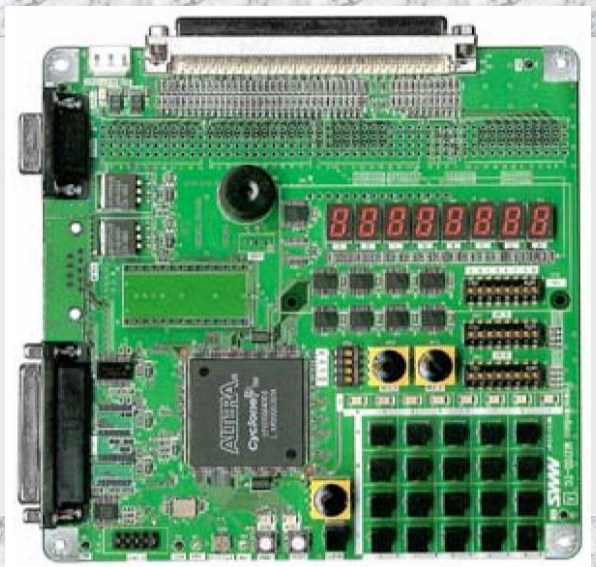
a	b	c	d	out
0	0	0	0	?
0	0	0	1	?
⋮				⋮
1	1	1	1	?

実験で使うFPGA

- （旧ボード）Altera EP1C6Q240C8
  - 5980ロジック・エレメント
    - 各LEは4入力LUTとレジスタで構成
  - 92160bitのRAM
  - 240ピンPQFPパッケージ

- （新ボード）Altera EP4C30F23I7N
  - 28848ロジック・エレメント
  - 540672bitのRAM
  - 66個の18x18マルチプライヤ
  - 484ピンFBGAパッケージ

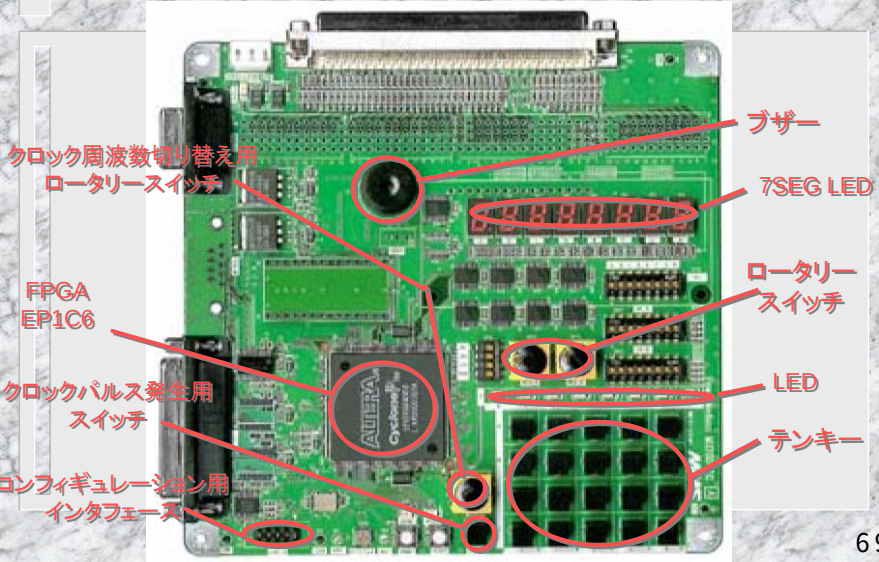
PowerMedusaボード(MU200-EC6S)



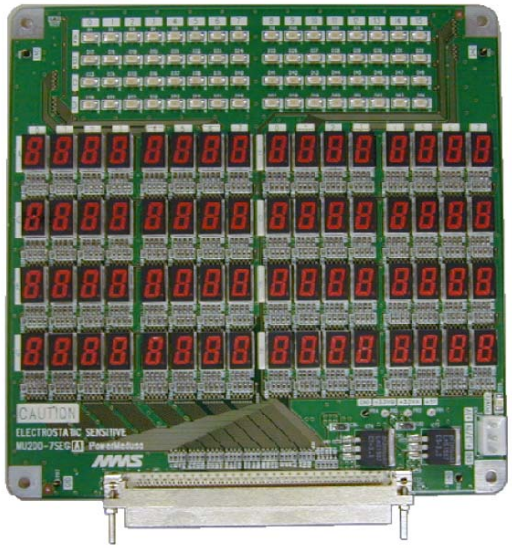
PowerMedusaボード(MU200-EC6S)

- トレーニング用ＦＰＧＡボード
  - FPGA Altera社 Cyclone EP1C6Q240C8
    - メモリはFPGA内部にSRAMブロックで構成
  - I/O
    - 入力
      - テンキー 20個
      - ロータリースイッチ 2個
    - 出力
      - LED 8個
      - 7SEG LED 8個
      - ブザー
  - その他
    - クロック発振器、クロックパルス発生スイッチ
    - リセットスイッチ

PowerMedusaボード(MU200-EC6S)

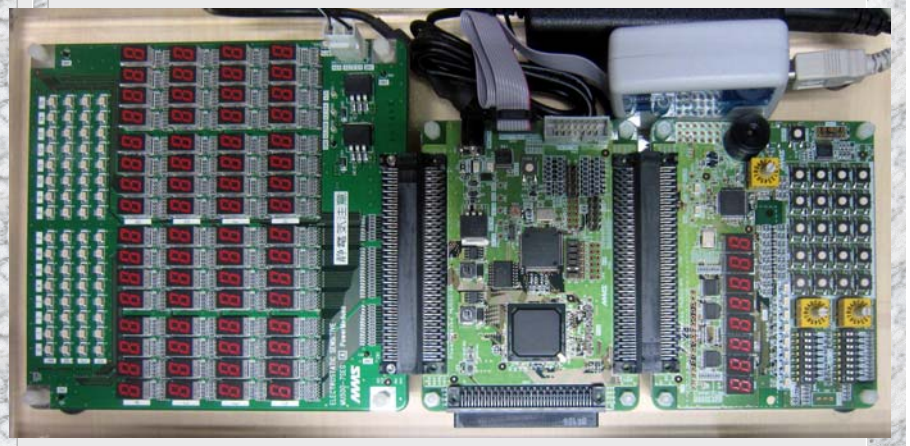


PowerMedusaボード(MU200-7SEG)



- 結果出力用ボード
  - 8個の7SEG LEDが8セット
  - 64個のLEDが1セット
  - 一度に1セットのみ書き換え可能
  - 書き換えない場合、値はホールドされる
- 設計が込み入ってきたら使うと便利

PowerMedusaボード(MU500-RX/RK/7SEG)



ボードの詳しい情報

- 実験Web上のマニュアル参照（印刷したのも少し用意あり）
  - FPGAの各ピンと入出力装置がどうつながっているか
    - ピンとLEDやブザーの間にはon/offスイッチがあります
  - 各スイッチの諸元はどうなっているか
    - 負論理のスイッチもあります
  - 確かどこか間違ってたと思いますので注意
    - 0,1,...7が1,2,...8になってたような気がします

座学の概要

1. ハードウェア実験の意味
2. SIMPLEアーキテクチャ
3. PowerMedusaボード
4. 実験の進め方

提出課題(1/2)

- 初回レポート: 個人単位, 4/16(木) 17:00締切
  - 1桁の16進数を7SEG LED上で0-Fで表示させる組み合わせ論理回路、及び、カウンタ回路の設計と実装
- 中間報告: グループ単位
  - レポート: 5/7(木) 17:00締切
    - 各種仕様書、考察等
    - 設計データを別途提出
  - デモ: 5/8(金) 8:45–12:00
    - 何らかの命令が動作している所を見せる
    - 一部の命令だけでも動作できる実装で間に合わせるのもあり

提出課題 (2/2)

- 最終報告: グループ単位
  - デモ: 5/28(木) 13:00-16:15
    - 設計したプロセッサの特徴の説明
      - 「SIMPLE/Bに比べてこんな点がすぐれているんですよ」というセールストークをして下さい
    - 応用プログラムの実行
  - レポート: 6/11(木) 17:00締切
    - 最終成果物のユーザーズマニュアル
    - SIMPLE/B基本仕様からの拡張および性能評価
    - 各コンポーネントの仕様書（最終版）、性能評価
    - 考察、感想
    - 設計データを別途提出

課題とツールについて

- HDLで設計してよいか
  - (最初の課題を除き) Verilog-HDL, VHDL 使用OK。
  - 相方とよく打ち合わせること。
- Megafunction (演算器等)を使ってよいか
  - OK。ただし部品を呼び出して使っただけの部分は評価対象外なので、他のところで工夫してそれなりにすごいものを作ること。
- いずれの場合も、どう設計したか、何を使ったか、レポートに明記すること。
- 単に動くだけではなく、スピード、回路規模、消費電力の最適化を目指すこと。

- 自宅で設計できる？
  - QuartusII Web Edition (ユーザ登録無料)を入れて使えます。
  - 最新版は旧ボードのCycloneに対応していないので注意。
- シミュレータはどこ？
  - QuartusII Ver.10からは別ツール(ModelSim)を使います。

コンテスト

- 「俺のプロセッサはすごいぜ！」ということを証明したいあなたに…
- データをソートする時間を競うコンテスト
  - データ
    - 16bitの符号付整数1024個
    - ランダム、昇順ソート済み、降順ソート済みの3種類
  - 時間の定義
    - 完了までのクロック・サイクル数×クロック周波数
    - 3種類のデータ各々の処理時間の平均値
- ぜひ参加して、昨年度の記録を破って下さい。
  （過去の記録と比較するなら旧ボードでの動作が必要。）

実験ホームページ

http://www.kuis.kyoto-u.ac.jp/ecs/isle/ の実験及演習３Ａ

- カレンダー
  - 大まかな作業内容
  - 3回の報告の日程、報告内容
- 資料
  - 初回講義スライド、SIMPLE仕様書、Power Medusa取扱説明書、CADのチュートリアル
  - よくある質問とその答え(FAQ)
  - Altera社のQuartusⅡの資料へのリンク
- 教員／TAの連絡先